



»Лекционен курс »ООП1 (Java)



Базови езикови конструкции



Въвеждащ Пример



Какво прави програмата?

```
import java.util.Scanner;

public class FirstProgram
{
    public static void main(String[] args)
    {
        System.out.println("Здравейте!");
        System.out.println("Ще събере две числа за Вас.");
        System.out.println("Въведете две цели числа на един ред:");

        int n1, n2;

        Scanner keyboard = new Scanner(System.in);
        n1 = keyboard.nextInt();
        n2 = keyboard.nextInt();

        System.out.println("Сумата на двете числа е");
        System.out.println(n1 + n2);
    }
}
```

Събира две цели числа



Изпълнение на програмата



Какъв е резултатът?

```
import java.util.Scanner;

public class FirstProgram
{
    public static void main(String[] args)
    {
        System.out.println("Здравейте!");
        System.out.println("Ще събере две числа за Вас.");
        System.out.println("Въведете две цели числа на един ред:");

        int n1, n2;

        Scanner keyboard = new Scanner(System.in);
        n1 = keyboard.nextInt();
        n2 = keyboard.nextInt();

        System.out.println("Сумата на двете числа е");
        System.out.println(n1 + n2);
    }
}
```

```
Здравейте!
Ще събере две числа за Вас.
Въведете две цели числа на един ред:
2 3
Сумата на двете числа е
5
```

Потребител

- » **Потребител**: човекът, който взаимодейства с програма
- » В определени случаи потребителят може да бъде програмистът, т.е. човекът, който е написал програмата
 - > Обикновено студентите често са програмисти и потребители
- » В разработването на реален софтуер програмистът и потребителят обикновено са различни хора

Въвеждащ пример



Каква е програмата?

```
import java.util.Scanner;

public class FirstProgram
{
    public static void main(String[] args)
    {
        System.out.println("Здравейте!");
        System.out.println("Ще събере две числа за Вас.");
        System.out.println("Въведете две цели числа на един ред:");

        int n1, n2;

        Scanner keyboard = new Scanner(System.in);
        n1 = keyboard.nextInt();
        n2 = keyboard.nextInt();

        System.out.println("Сумата на двете числа е");
        System.out.println(n1 + n2);
    }
}
```

Императивна



Структура на програмата

```
import java.util.Scanner;
```

Получава класа Scanner от пакета (библиотека) java.util

```
public class FirstProgram {  
    public static void main(String[] args,
```

Име на класа – по ваш избор. Тази програма трябва да е във файл с име FirstProgram.java

```
    {  
        System.out.println("Здравейте!");
```

```
        System.out.println("Ще събера две числа за вас.");
```

```
        System.out.println("Въведете две цели числа на един ред:");
```

Изпраща изход на екран

```
        int n1, n2;
```

Казва, че n1 и n2 са променливи, които съдържат цели числа (int)

```
        Scanner keyboard = new Scanner(System.in);
```

```
        n1 = keyboard.nextInt();
```

```
        n2 = keyboard.nextInt();
```

Чете цяло число от клавиатурата

Прочита програмата за въвеждане от клавиатурата

```
        System.out.println("Сумата на двете числа е");
```

```
        System.out.println(n1 + n2);
```

```
    }
```

```
}
```

Обобщение

```
import java.util.Scanner;

public class FirstProgram
{
    public static void main(String[] args)
    {
        System.out.println("Здравейте!");
        System.out.println("Ще събера две числа за Вас.");
        System.out.println("Въведете две цели числа на един ред:");

        int n1, n2;

        Scanner keyboard = new Scanner(System.in);
        n1 = keyboard.nextInt();
        n2 = keyboard.nextInt();

        System.out.println("Сумата на двете числа е");
        System.out.println(n1 + n2);
    }
}
```

Не се притеснявайте, ако някои от детайлите на програмата не са напълно ясно за това първо четене.

Императивно програмиране

» Алгоритъм:

- > Подход за изчисляване на търсене стойности от дадени такива, . . . който използва постъпково изпълнение на елементарни обработващи операции

» Императивна програма:

- > Алгоритми, описани посредством обръщания (достъп) към стойности на променливи
- > Четене и промяна на стойности

*Императивно програмиране е ориентирано
към описание на алгоритми*



Императивна програма



Основни части на една императивна програма?

```
import java.util.Scanner;

public class FirstProgram
{
    public static void main(String[] args)
    {
        System.out.println("Здравейте!");
        System.out.println("Ще събера две числа за Вас.");
        System.out.println("Въведете две цели числа на един ред:");

        int n1, n2;

        Scanner keyboard = new Scanner(System.in);
        n1 = keyboard.nextInt();
        n2 = keyboard.nextInt();

        System.out.println("Сумата на двете числа е");
        System.out.println(n1 + n2);
    }
}
```

Извежда резултати

Получава входни данни

Обработва данни

Императивно програмиране

» Базови концепции:

- > **Променливи:** притежават стойности, които се променят посредством прилагане на оператори
- > **Оператори:** Служат за достъп до стойностите на променливите (четене, промяна, извеждане на стойностите)

» Базов метод за структуриране на императивното програмиране:

- > **Процедури (функции, методи):** Частични алгоритми, представени като оператори на езика

Императивно програмиране

- » Данни (променливи) и обработващи алгоритми (процедура, функция, метод) **са разделени** структури в императивното програмиране
- » За сравнение с обектно-ориентираното:
 - > **Клас** = единство от данни и обработващи алгоритми

*Java: основно предназначение обектно-ориентирано програмиране.
Обаче, императивното програмиране е възможно.*

Компоненти



От колко компонента (класа) се състои програмата?

```
import java.util.Scanner;
```

```
public class FirstProgram
```

3

```
{
```

```
    public static void main(String[] args)
```

```
{
```

```
        System.out.println("Здравейте!");
```

2

```
        System.out.println("Ще събере две числа за Вас.");
```

```
        System.out.println("Въведете две цели числа на един ред:");
```

```
        int n1, n2;
```

1

```
        Scanner keyboard = new Scanner(System.in);
```

```
        n1 = keyboard.nextInt();
```

```
        n2 = keyboard.nextInt();
```

```
        System.out.println("Сумата на двете числа е");
```

```
        System.out.println(n1 + n2);
```

```
    }
```

```
}
```

3



12

Компоненти на програмата



Защо възможно?

Разделено компилиране

Дефиниран от потребителя
клас

```
class FirstProgram {  
    ...  
}
```

File: FirstProgram.java

```
class Scanner {  
    ...  
}
```

File: Scanner.java

```
class System {  
    ...  
}
```

Java API(Application Programming Interface)
(стандартни библиотеки)

Интерфейси между компонентите

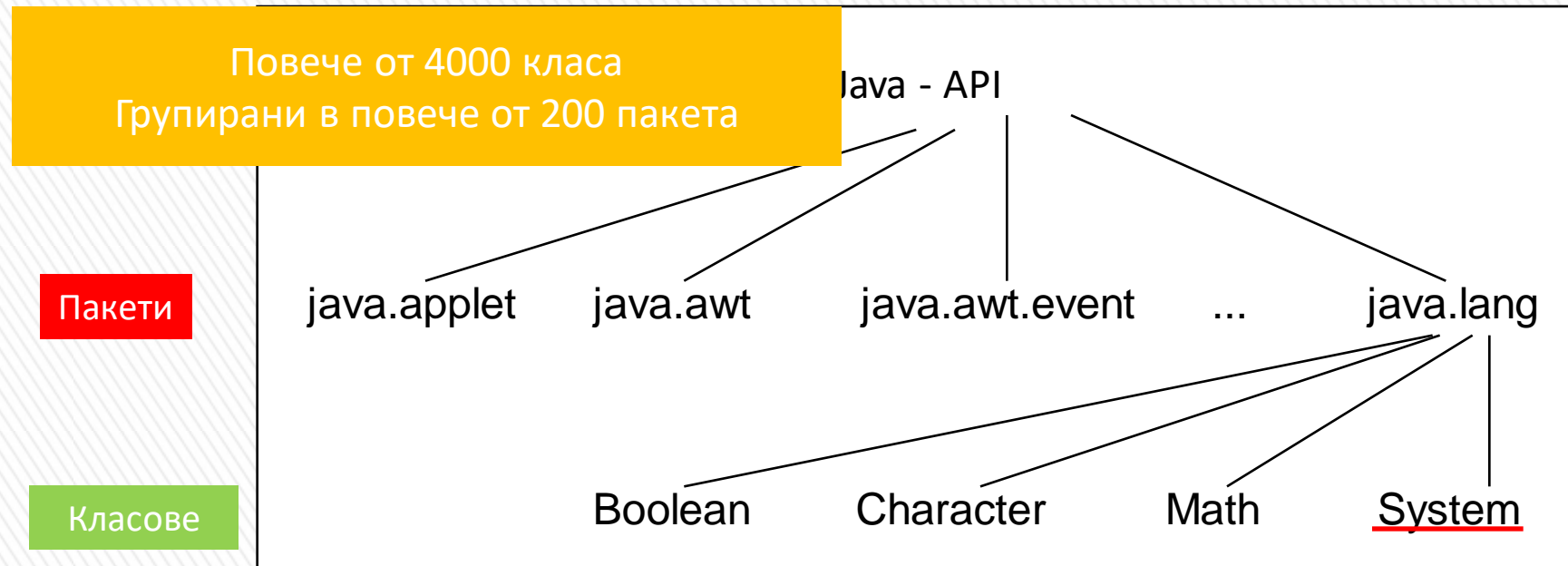
```
class FirstProgram {  
  
    public static void main (String [] args) {  
        ...  
        keyboard = Scanner...;  
        System.out.print(" deg F is ");  
        ...  
    }  
}
```

Java API

```
class System {  
    ...  
}
```

```
class Scanner {  
    ...  
}
```

Организация на Java-API



Пакети:
Сбирка от класове

Класове:
Софтуерни компоненти

Име на пакет (напр. java.awt.event) отразява Directory-Names:
→ /java/awt/event

Спецификация на пакети

Java™ Platform, Standard Edition 7 API Specification

This document is the API specification for the Java™ Platform, Standard Edition.

See: Description

Packages	
Package	Description
java.applet	Provides the classes necessary to create and run applets.
java.awt	Contains all of the classes for creating and running the user interface of the application.
java.awt.color	Provides classes for color spaces.
java.awt.datatransfer	Provides interfaces and classes for transferring data from and to the system clipboard.
java.awt.dnd	Drag and Drop is a direct manipulation interface for moving and copying graphical elements in the GUI.
java.awt.event	Provides interfaces and classes for defining and processing events.
java.awt.font	Provides classes and interface relating to text layout and font rendering.
java.awt.geom	Provides the Java 2D classes for defining and manipulating geometric shapes.
java.awt.im	Provides classes and interfaces for the input method framework.
java.awt.im.spi	Provides interfaces that enable the development of input methods that can be used with any Java runtime environment.
java.awt.image	Provides classes for creating and modifying images.
java.awt.image.renderable	Provides classes and interfaces for producing rendering-independent images.
java.awt.print	Provides classes and interfaces for a general printing API.
java.beans	Contains classes related to developing beans -- components based on the JavaBeans™ architecture.
java.beans.beancontext	Provides classes and interfaces relating to bean context.
java.io	Provides for system input and output through data streams, serialization and the file system.
java.lang	Provides classes that are fundamental to the design of the Java programming language.
java.lang.annotation	Provides library support for the Java programming language annotation facility.
java.lang.instrument	Provides services that allow Java programming language agents to instrument programs running on the JVM.
java.lang.invoke	The java.lang.invoke package contains dynamic language support provided directly by the Java core class libraries and virtual machine.
java.lang.management	Provides the management interfaces for monitoring and management of the Java virtual machine and other components in the Java runtime.
java.lang.ref	Provides reference-object classes, which support a limited degree of interaction with the garbage collector.

Пакет: java.lang

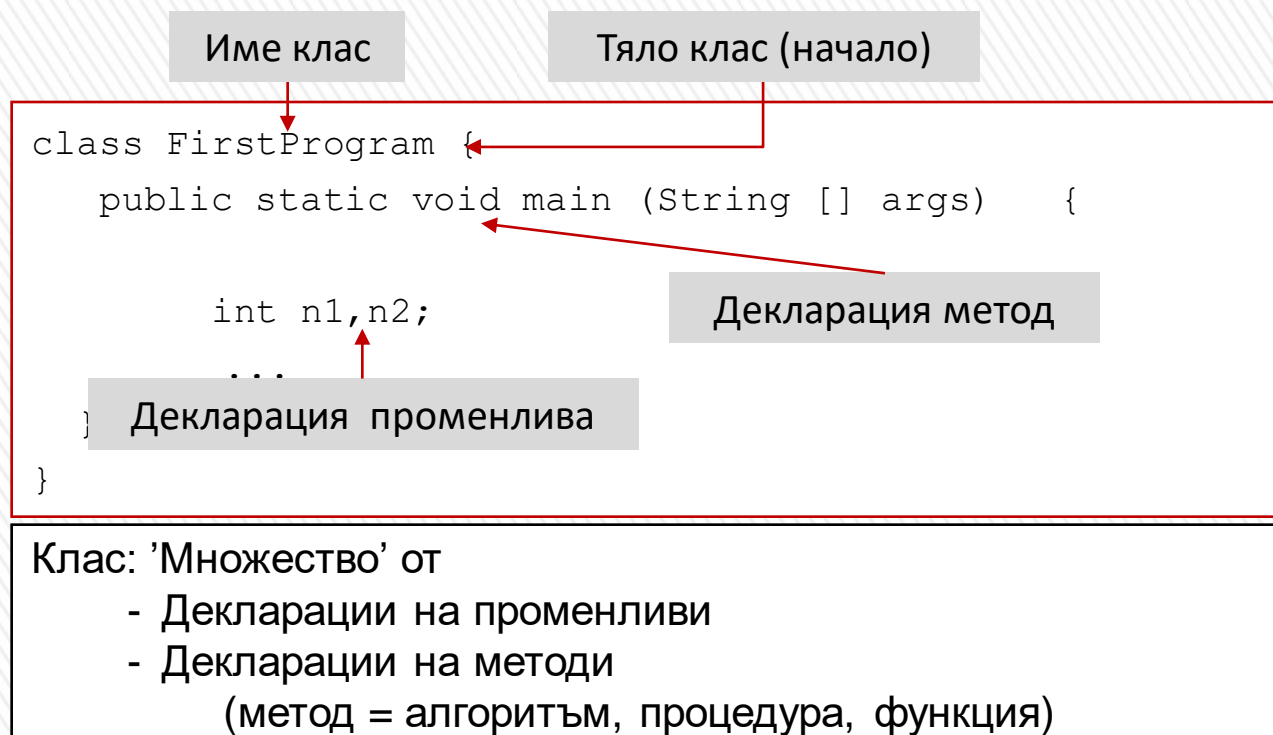
- Автоматично се доставя от системата
- Не е необходимо да се импортира

Java програми

- » Принципно два вида Java програми:
 - > Приложения
 - > Аплети
- » Могат да съдържат повече от един клас
- » При приложенията точно един клас трябва да съдържа дефиниция на метод с име **main**
 - > Активира се когато започне изпълнението на приложението

Структура на Java-програми

Клас = базов компонент



За общата програма: един метод `main()` → там започва обработката !



Благодаря за вниманието!